

# Practical approximate projection schemes in greedy signal space methods

Chris Garnatz,<sup>\*</sup> Xiaoyi Gu,<sup>†</sup> Alison Kingman,<sup>‡</sup> James LaManna,<sup>§</sup> Deanna Needell,<sup>¶</sup> Shenyinying Tu,<sup>||\*,\*</sup>

September 5, 2014

## Abstract

Compressive sensing (CS) is a new signal acquisition paradigm which shows that far fewer samples are required to reconstruct sparse signals than previously thought. Although most of the literature focuses on signals sparse in a fixed orthonormal basis, recently the Signal Space CoSaMP (SSCoSaMP) greedy method was developed for the reconstruction of signals compressible in arbitrary redundant dictionaries. The algorithm itself needs access to approximate sparse projection schemes, which have been difficult to obtain and analyze. This paper investigates the use of several different projection schemes and catalogs for what types of signals each scheme can successfully be utilized. In addition, we present novel hybrid projection methods which outperform all other schemes on a wide variety of signal classes.

## 1 Introduction

Compressive Sensing (CS) is an emerging field that seeks to optimize signal acquisition and reconstruction. CS relies on the fact that most real world signals are themselves sparse or compressible with respect to some fixed basis. Traditionally, this basis is assumed to be orthonormal. However, most signals in practice are instead sparse with respect to non-orthonormal, highly overcomplete dictionaries. CS has only recently begun to develop technologies for this framework, and we investigate some of them in this paper.

We will use the following notation to frame the fundamental problem CS seeks to answer. The signal in question is  $\mathbf{x} \in \mathbb{C}^n$ . The measurement vector  $\mathbf{y} \in \mathbb{C}^m$  is related to the signal by the measurement matrix  $\mathbf{A} \in \mathbb{C}^{m \times n}$  and additive noise  $\mathbf{e} \in \mathbb{C}^m$ , where  $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$  and  $m \ll n$ . We impose a sparsity condition on  $\mathbf{x}$  by the equation  $\mathbf{x} = \mathbf{D}\boldsymbol{\alpha}$ , where  $\boldsymbol{\alpha} \in \mathbb{C}^d$  with  $|\text{supp}(\boldsymbol{\alpha})| \leq k \ll n$  is a  $k$ -sparse representation of the signal  $\mathbf{x}$  with respect to the dictionary  $\mathbf{D} \in \mathbb{C}^{n \times d}$ . Note that when  $d = n$  and  $\mathbf{D} = \mathbf{I}_n$  is the identity, then we have the case where  $\mathbf{x}$  is itself sparse.

One can recover a sparse vector from its measurements by solving an  $\ell_0$ -minimization problem which simply searches for the sparsest vector which yields the same measurements. Although this is an NP-hard problem, the seminal work of Candès, Romberg and Tao [6, 4] shows that a sparse vector  $\mathbf{x}$  can be recovered using the relaxed  $\ell_1$ -minimization method from the measurement vector  $\mathbf{y}$  as long as the measurement matrix  $\mathbf{A}$  satisfies a condition known as the Restricted Isometry Property (RIP). To say  $\mathbf{A}$  satisfies the (RIP) of order  $k$  is to say the following: for a constant  $\delta_k \in (0, 1)$ , the condition

$$(1 - \delta_k) \|\mathbf{x}\|_2^2 \leq \|\mathbf{A}\mathbf{x}\|_2^2 \leq (1 + \delta_k) \|\mathbf{x}\|_2^2 \quad (1.0.1)$$

<sup>\*</sup>Pomona College, Claremont CA 91711

<sup>†</sup>Univ. of California, Los Angeles CA 90095

<sup>‡</sup>Harvey Mudd College, Claremont CA 91711

<sup>§</sup>Pitzer College, Claremont CA 91711

<sup>¶</sup>Claremont McKenna College, Claremont CA 91711 dneedell@cmc.edu

<sup>||</sup>Univ. of California, Los Angeles CA 90095

<sup>\*\*</sup>This work was supported by NSF grant DMS-1045536, NSF CAREER grant #1348721, and the Alfred P. Sloan Fellowship.

holds for all  $\mathbf{x}$  with  $|\text{supp}(\mathbf{x})| \leq k$ . Finding a measurement matrix  $\mathbf{A}$  that satisfies the RIP for a sparsity level  $k$  is a nontrivial endeavor. Fortunately, if the entries of  $\mathbf{A}$  are drawn from a subgaussian distribution and we use  $m \geq Ck/\log(n)$  measurements, then  $\mathbf{A}$  satisfies the RIP with high probability [5, 15]. Similar results hold for subsampled Discrete Fourier Transform (DFT) matrices and others with a fast-multiply [15, 14]. Thus a measurement matrix  $\mathbf{A}$  with sufficient properties for signal recovery can be practically constructed. Greedy approaches such as OMP [16, 17], ROMP [13], CoSaMP [12], and IHT [1] which identify elements of the signal support iteratively also provide robust reconstruction guarantees similar to those of the  $\ell_1$  approach. We point the reader to these references for details about each of the CS algorithms.

## 2 Signal Space Methods

Existing CS literature has largely addressed the scenarios where  $\mathbf{x}$  is sparse or compressible with respect to an orthonormal dictionary  $\mathbf{D}$ . When  $\mathbf{D}$  is orthonormal, these methods simply recover  $\boldsymbol{\alpha}$ , which naturally yields  $\mathbf{x}$ . Most signals in practical applications, however, are compressible instead with respect to a highly overcomplete dictionary such as an oversampled DFT, Gabor frame, or many of the redundant frames used in image processing. Unfortunately, when  $\mathbf{D}$  is not orthonormal, these CS methods fail both theoretically and empirically.

In [8, 7], Davenport et. al. propose a new method called Signal Space CoSaMP (SSCoSaMP), specifically designed to recover the signal  $\mathbf{x}$  when  $\mathbf{D}$  is not guaranteed to be orthonormal. They utilize a generalization of the RIP, the  $D$ -RIP criterion introduced in [2] for the  $\ell_1$ -analysis approach, to ensure signal recovery. In contrast to other CS algorithms, SSCoSaMP doesn't return  $\hat{\boldsymbol{\alpha}}$  to obtain  $\hat{\mathbf{x}}$ , but instead opts for a "signal-focused" approach and attempts to directly calculate and return  $\hat{\mathbf{x}}$ .

The two key steps within SSCoSaMP require finding the best  $k$ -sparse approximation of a vector  $\mathbf{z}$  with respect to the arbitrary dictionary  $\mathbf{D}$ :

$$\Omega_{\text{opt}} := \underset{\Lambda: |\Lambda|=k}{\operatorname{argmin}} \|\mathbf{z} - \mathcal{P}_{\Lambda}\mathbf{z}\|_2,$$

where  $\mathcal{P}_{\Lambda}$  denotes the projection onto the span of the columns of  $\mathbf{D}$  indexed by  $\Lambda$ . Unfortunately, finding the support of such an optimal sparse projection vector is an NP-hard problem itself. Instead, in the interest of computational efficiency, SSCoSaMP computes a *near-optimal* approximation [7, 10] of the support via a simpler CS algorithm, which is noted by the function  $\mathcal{S}_{\mathbf{D}}$ . Stunningly, SSCoSaMP is able to accurately recover signals with great success even when  $\mathbf{D}$  is very overcomplete and redundant, using a standard CS method like OMP, CoSaMP or  $\ell_1$ -minimization for the near-optimal projection  $\mathcal{S}_{\mathbf{D}}$ . The SSCoSaMP algorithm is described in Algorithm 1, see [7] for a detailed description.

---

### Algorithm 1 Signal-Space CoSaMP (SSCoSaMP)

---

**Input:**  $\mathbf{A}$ ,  $\mathbf{D}$ ,  $\mathbf{y}$ ,  $k$ , stopping criterion  
**Initialize:**  $\mathbf{r} = \mathbf{y}$ ,  $\mathbf{x}_0 = 0$ ,  $\ell = 0$ ,  $\Gamma = \emptyset$   
**while** not converged **do**  
    **Proxy:**  $\tilde{\mathbf{v}} = \mathbf{A}^* \mathbf{r}$   
    **Identify:**  $\Omega = \mathcal{S}_{\mathbf{D}}(\tilde{\mathbf{v}}, 2k)$   
    **Merge:**  $T = \Omega \cup \Gamma$   
    **Update:**  $\tilde{\mathbf{w}} = \operatorname{argmin}_{\mathbf{z}} \|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2 \quad \text{s.t.} \quad \mathbf{z} \in \mathcal{R}(\mathbf{D}_T)$   
    **Prune:**  $\Gamma = \mathcal{S}_{\mathbf{D}}(\tilde{\mathbf{w}}, k)$   
     $\mathbf{x}_{\ell+1} = \mathcal{P}_{\Gamma} \tilde{\mathbf{w}}$   
     $\mathbf{r} = \mathbf{y} - \mathbf{A}\mathbf{x}_{\ell+1}$   
     $\ell = \ell + 1$   
**end while**  
**Output:**  $\hat{\mathbf{x}} = \mathbf{x}_{\ell}$

---

SSCoSaMP is a state of the art algorithm, but in its present form, its recovery performance varies greatly with what CS algorithm is used for the near-optimal projection and with the structure of the sparse coefficient vector

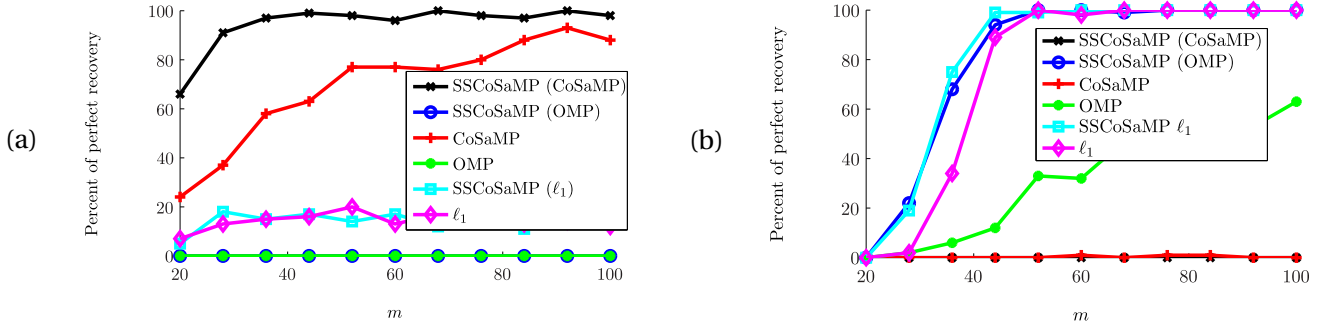


Figure 1: Percent of perfect recovery for different SSSoSaMP variants when the nonzero entries in  $\alpha$  are clustered together (left) and when the nonzero entries in  $\alpha$  are well-separated (right).

$\alpha$ . This phenomenon is highly evident in the simulation results presented by the authors of [7], which we recreate in Figure 1, which shows percent of perfect recovery as a function of the number of measurements  $m$ .<sup>1</sup> Here, the dictionary  $D \in \mathbb{C}^{n \times d}$  is a  $4 \times$  overcomplete DFT and  $A \in \mathbb{R}^{m \times n}$  has standard normal entries. For all 100 trials,  $k = 8$ ,  $n = 256$ , and  $d = 1024$ . Here and throughout this paper, the algorithm in parentheses following “SSCoSaMP” is the algorithm used for the approximate projection  $\mathcal{S}_D$  in the identify and prune steps.

## 2.1 Contribution

As observed in Figure 1, the success of SSSoSaMP depends heavily on both the approximate projection used as well as the structure of the signal; more interesting, none of the methods work well on both signal types. Unfortunately, this phenomenon is not well understood theoretically and even experimentally. This motivates the two main points of focus for this paper. First, we investigate how variants of SSSoSaMP perform on sparse vectors of varying structures. We rigorously test each individual algorithm on a variety of signal types in order to precisely identify where some algorithms succeed and where others fall short, thus cataloging which methods should be used on which signal classes. Secondly, we develop novel CS algorithm variants with the ability of allowing reconstruction over a much wider range of signal structures. This aims to remove the need to select specific CS methods depending on the structure of the signal of interest. Our methods improve upon all other existing approaches in this regard.

## 3 Empirical Investigation

The stark contrast in the recovery performance between the SSSoSaMP variants in the clustered and well-separated cases is the main motivation for the rest of this paper. We aim to explore why this sharp rift exists and exactly how far it extends. We reason that the clustered and well-separated cases represent two extremes – a vast middle ground lay between these two poles that has yet to be explored. To shed some light on this unknown, we create multiple classes of sparse vectors that lay somewhere between these two extremes. In addition to the *clustered* and *spread* signal types of Figure 1, we also define the following classes of sparse signals:

- **Hybrid:** a block of  $k/2$  nonzeros and an additional  $k/2$  nonzeros at least 8 indices away from all other nonzeros
- **C Clusters:**  $C$  blocks of  $k/C$  nonzeros with at least one zero between the blocks
- **Alternating:** a block of alternating nonzero and zero entries, for a total length of  $2k - 1$  indices
- **Pair Spread:**  $k/2$  pairs of nonzeros where the nonzeros in each pair are 4 indices apart

<sup>1</sup>We define perfect recovery as signal to noise ratio  $20 \log(\|\mathbf{x}\|_2 / \|\mathbf{x} - \hat{\mathbf{x}}\|_2)$  greater than 100db.

**Experimental parameters.** For all of the sparse vectors described above, the actual placement of the nonzero entries is random every trial. We only ensure the the random placement adheres to our general structure. Additionally, in every trial we use a  $4\times$  overcomplete Discrete Fourier Transform (DFT) dictionary with dimensions  $256 \times 1024$ . The measurement matrix  $\mathbf{A}$  is an  $m \times 256$  matrix with standard normal entries. Unless otherwise stated, the sparsity level is held at  $k = 8$ . Recovery results for different SS-CoSaMP variants can be seen in Table 1 at the end of Section 5.

The first, and most telling, sparse vector on which we test the SS-CoSaMP algorithms is the hybrid signal, seen in Figure 2. We see that neither the SS-CoSaMP variants nor the classical CS algorithms recover the signal with any reasonable success. While some algorithms do well in a clustered case and others do well with a spread signal, this test shows that none of the algorithms do well when the signal is part clustered and part well-separated.

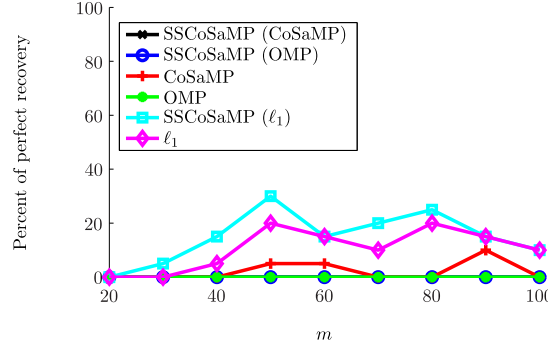


Figure 2: Conventional SS-CoSaMP and CS algorithms on recovering a sparse vector with a hybrid sparse support: a block of  $k/2$  nonzeros with the remaining  $k/2$  nonzeros spaced at least 8 slots apart from all other nonzeros.

The following test, seen in Figure 3 (top), summarizes the transition of the performance in the algorithms from a clustered signal to a well-separated one. The  $x$ -axis now represents the number of zeros between each nonzero entry in a signal. For example, at zero the support of the signal consists of a single cluster of indices, while at a value of 10 separations there are 10 zeros between each nonzero entry. The measurements and sparsity level are held at a constant value,  $m = 100$  and  $k = 8$ , respectively.

At a separations value of zero we see SS-CoSaMP (CoSaMP) and CoSaMP achieve 100% perfect recovery, while the other algorithms all perform poorly. As the separations increase to the range of three to four there is a steep drop in the performance of SS-CoSaMP (CoSaMP) and CoSaMP. One interesting note is that CoSaMP doesn't drop as fast as SS-CoSaMP (CoSaMP) does. From five separations onwards SS-CoSaMP (CoSaMP) and CoSaMP have no success in recovering the signal. When the separations increase to five, SS-CoSaMP ( $\ell_1$ ), SS-CoSaMP (OMP), and  $\ell_1$  start to perform very well. This shows us that in this case the nonzeros need to be at least five spaces apart to be spread enough for these algorithms to recover the signal consistently. We attribute this point to the periodic correlation of columns in the dictionary matrix  $\mathbf{D}$  that is  $4\times$  overcomplete. This makes it very difficult for the algorithms to recover the signal at four separations. Indeed, this required separation is also evident in the super-resolution setting [3]. But, as the separations increase past four, the SS-CoSaMP ( $\ell_1$ ), SS-CoSaMP (OMP), and  $\ell_1$  algorithms can recover the signal because they are no longer affected by the periodic correlation of the columns.

The test seen in Figure 3 (bottom) is very similar to the previous test. However, the separation value now represents the number of zeros between two clusters of size  $k/2$ . Again, sparsity and measurements are fixed at  $k = 8$  and  $m = 100$ , respectively. As expected CoSaMP and SS-CoSaMP (CoSaMP) perform well for low separation but as the separation between the two clusters increases SS-CoSaMP (CoSaMP) starts to perform very poorly. CoSaMP stays around 90% of perfect recovery for all separation values; it is the only consistent algorithm in this case. SS-CoSaMP ( $\ell_1$ ), SS-CoSaMP (OMP), OMP, and  $\ell_1$  never do well in this test. This result is expected because we have already observed that these algorithms don't perform well when the signal has a clustered type of structure.

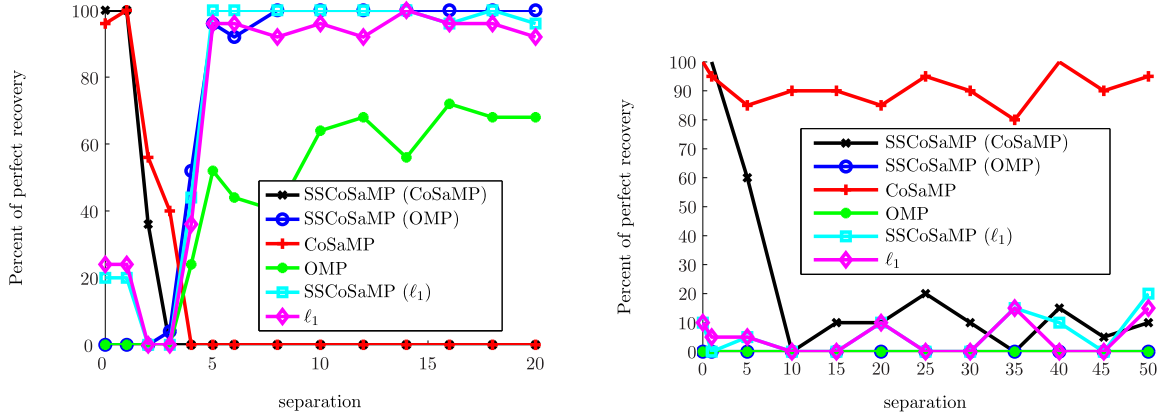


Figure 3: Conventional SSCoSaMP and CS algorithms on recovering a sparse vector. On the left the separations represent the number of zeros between each nonzero entry, and on the right the separations represent the number of zeros between two clusters each of size  $k/2$ . Measurements and sparsity are held constant at  $m = 100$  and  $k = 8$ , respectively.

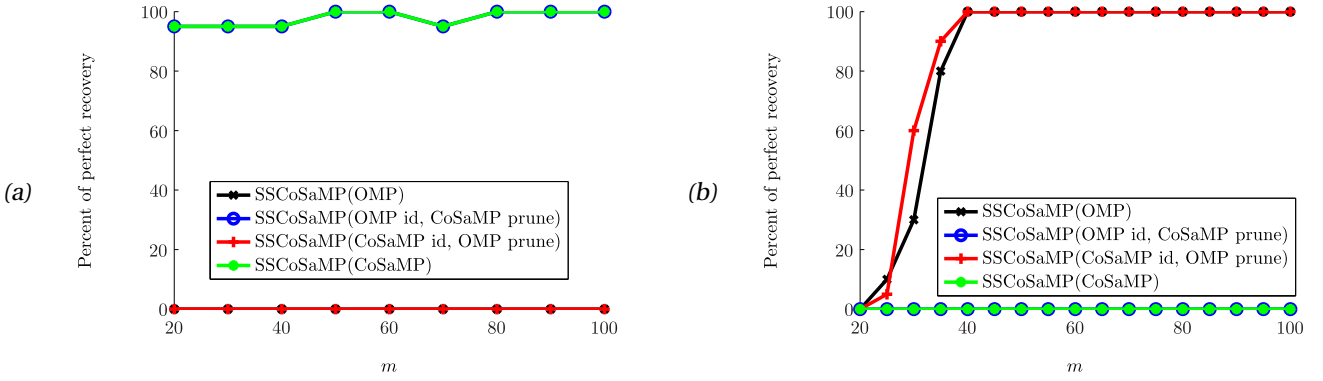


Figure 4: *SSCoSaMP* variants with differing identification and prune steps, recovery performance on a sparse vector with a single cluster of nonzeros (left) and a sparse vector with well-separated nonzeros (right).

To better understand the limitations of *SSCoSaMP*, we also examine the inner steps of *SSCoSaMP* that employ a classical CS method, the prune and identify steps. In Figure 4, we create two new *SSCoSaMP* variants. The first uses OMP to identify the near optimal  $2k$ -sparse support for the proxy vector and CoSaMP to prune the support vector to its best  $k$ -sparse approximation. The second variant uses CoSaMP to identify and OMP to prune.

We test these two new variants on the two extreme cases of sparse vectors, and the results were definitive. The two new variants behave almost exactly like the pure *SSCoSaMP* variant whose prune step matches their own. From this, we conclude that the algorithm used to prune determines the overall performance of the algorithm.

## 4 Algorithmic Variations

### 4.1 NOMP

Orthogonal Matching Pursuit (OMP) as described in [11], is a greedy algorithm that uses the signal,  $\tilde{\mathbf{v}} = \mathbf{A}^* \mathbf{A}(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}})$  as a proxy for the true desired signal  $\mathbf{x}$ . OMP constructs the support set by iteratively selecting the coordinate corresponding to the largest coefficient of the proxy  $\tilde{\mathbf{v}}$ . Call this coordinate  $\lambda$ . OMP iterates  $k$  times, where  $k$  is the sparsity level. In each iteration, the support and residual are updated. Once the support of the signal  $\Gamma$  is found, the signal is estimated using a simple least squares problem as  $\hat{\mathbf{x}} = \mathbf{A}_{\Gamma}^{\dagger} \mathbf{y}$ . However, past experiments show that OMP only successfully recovers well-separated signals in the signal space when  $\mathbf{D}$  is an overcomplete dictionary.

Here, we present a simple modification in order to recover signals having a variety of structures which are sparse in dictionaries with correlation patterns like the overcomplete DFT.

The variation of OMP we propose here is deemed Neighborly Orthogonal Matching Pursuit (NOMP). In each iteration of OMP, only one coordinate is added to the support set. In each iteration of NOMP, a  $w$ -window of coordinates is added to the support. That is, a cluster of  $w$  adjacent coordinates are added to  $\Gamma$ . The coordinate corresponding to the largest coefficient,  $\lambda$ , is at the center of the window. If  $w$  is odd, we select an equal number of adjacent coordinates from either side of the  $\lambda$ . If  $w$  is even, choose the last coordinate by selecting the largest coordinate on the edge of the window. In order to prevent a support set that is larger than the number of measurements  $m$ , we ensure that  $w \leq \frac{m}{k}$ . NOMP is described in Algorithm 2.

NOMP's method of constructing the support set is theoretically similar to the  $\epsilon$ -OMP method of [9].  $\epsilon$ -OMP assumes the use of a highly coherent dictionary and takes advantage of the correlated columns in  $\mathbf{D}$ .  $\epsilon$ -OMP adds additional coordinates to the support set if these coordinates are correlated with the largest one. That is,  $\epsilon$ -OMP uses the  $\epsilon$ -extension defined in [10], given by:

$$\text{ext}_{\epsilon,2}(\Gamma) = \left\{ i : \exists j \in \Gamma, \frac{|\langle d_i, d_j \rangle|^2}{\|d_i\|_2^2 \|d_j\|_2^2} \geq 1 - \epsilon^2 \right\} \quad (4.1.1)$$

This is an alternative way of extending the support set that would have been determined in traditional OMP.

When  $\mathbf{D}$  is an overcomplete DFT dictionary, since nearby columns in  $\mathbf{D}$  are correlated,  $\epsilon$ -OMP essentially adds coordinates neighboring  $\lambda$  to the support set. However, it may be difficult to tune the parameter  $\epsilon$  in order to select the desired number of nearby neighbors. For NOMP, one can always reliably set  $w = m/k$  if one does not know the structure of the dictionary.

NOMP, however, is designed only for a specific class of dictionaries whose columns have correlated neighbors and is therefore less generalized than  $\epsilon$ -OMP. NOMP explicitly adds a specified number of additional coordinates adjacent to the largest one. Therefore, we will know the exact size of the resulting support set and can ensure  $|\Gamma| \leq m$ . If there is knowledge of the approximate size of any clusters in the signal that needs to be recovered, we can change  $w$  as needed.

The main difference between NOMP and  $\epsilon$ -OMP is in the update step.  $\epsilon$ -OMP does not use the  $\epsilon$ -extension when updating the estimated signal. The extension is only used in the least squares step at the end of the algorithm. NOMP, on the other hand, includes the neighbors in the support in the update step and in the least squares step. It turns out that this may lead to a difference in performance.

---

**Algorithm 2** Neighborly Orthogonal Matching Pursuit (NOMP)

---

**input:** Measurement matrix  $\mathbf{A}$ , dictionary  $\mathbf{D}$ , measurement vector  $\mathbf{y} = \mathbf{A}\mathbf{x}$  where  $\mathbf{x} = \mathbf{D}\boldsymbol{\alpha}$ , sparsity level  $k$ , number of measurements  $m$ , window size  $w \leq m/k$

**initialize:** Let support set  $\Gamma = \emptyset$ , residual  $\mathbf{r} = \mathbf{y}$  and counter  $\ell = 0$ . Let  $\Phi = \mathbf{A}\mathbf{D}$ .

**while**  $\ell < k$  **do**

**proxy:**  $\tilde{\mathbf{v}} = \Phi^* \mathbf{r}$

**identify:** Select largest coordinate  $\lambda$  of  $\tilde{\mathbf{v}}$  and set  $T = \{\lambda\}$ . Select the  $\frac{w-1}{2}$  coordinates to the left and right of  $\lambda$ . Add these coordinates to  $T$ .

**update:**  $\Gamma = \Gamma \cup T$   
 $\tilde{\boldsymbol{\alpha}} = \arg\min_{\mathbf{z}} \|\mathbf{y} - \Phi_{\Gamma} \mathbf{z}\|_2$   
 $\mathbf{r} = \mathbf{y} - \Phi_{\Gamma} \tilde{\boldsymbol{\alpha}}$   
 $\ell = \ell + 1$

**end while**

**output:** Recovered coefficient vector,  $\hat{\boldsymbol{\alpha}} = \mathbf{A}_{\Gamma}^{\dagger} \mathbf{y}$   
Recovered signal,  $\hat{\mathbf{x}} = \mathbf{D}\hat{\boldsymbol{\alpha}}$

---

Our next experiments demonstrate the recovery performance of NOMP benchmarked against  $\epsilon$ -OMP, traditional compressed sensing algorithms and their SSCoSAMP variants. In all experiments,  $k = 8$ ,  $n = 256$ ,  $d = 1024$

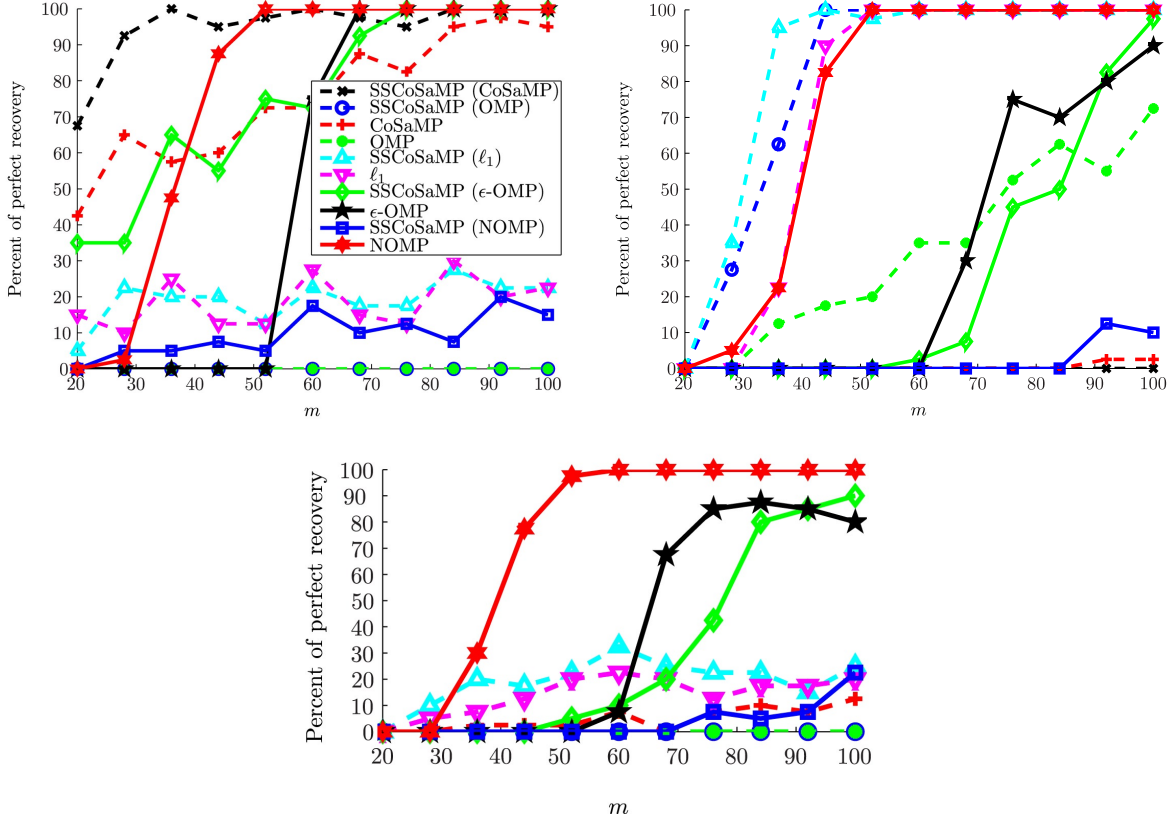


Figure 5: Percent of perfect recovery for clustered signals (left), well-separated signals (right) and hybrid signals (bottom).

and  $\mathbf{D}$  is a  $4 \times$  overcomplete DFT dictionary. In Figure 5,  $m$  is incremented from 20 to 100 by 8 and 40 trials are conducted for each value of  $m$ . For these tests, the maximum window size is  $w = 6$ . Figure 5 shows experiments on the recovery of clustered signals, well-separated signals and hybrid signals. NOMP has an excellent performance and is the only algorithm to have 100% perfect recovery for all three signals. Furthermore, it reaches 100% perfect recovery by about  $m = 50$  measurements.  $\epsilon$ -OMP has a decent performance when recovering all three signals. For these tests,  $\epsilon = 0.9539$ . This value is chosen because when using a  $4 \times$  overcomplete DFT dictionary, this value corresponds to the selection of six columns. Thus, we can fairly compare NOMP to  $\epsilon$ -OMP. NOMP's out-performance of  $\epsilon$ -OMP in these cases highlights the importance of using the neighbors in the update step, and of course  $\epsilon$ -OMP could also be modified in this way. We focus on NOMP because of its simplicity in this context.

Figures 6 and 7 show tests on NOMP and traditional compressed sensing algorithms using a variety of specific signal variants. For all of these tests, the number of measurements is fixed at  $m = 100$ . The figure captions provide a description of the type of signal tested. Observe that NOMP and  $\epsilon$ -OMP have the best performance. Figure 7 (right) shows NOMP's success when the sparsity of the signal is increased.

From these experiments, it is clear that NOMP is the algorithm with the best performance when recovering a variety of signal types, which is not surprising since it is designed specifically for this type of dictionary. Figure 8 shows the percent of perfect recovery of NOMP and  $\epsilon$ -OMP with varying values of  $\epsilon$ . All algorithms are tested on a hybrid signal since Figure 5 shows that NOMP is the only algorithm to successfully recover this type of signal. Figure 8 reveals that it is important to select the correct value of  $\epsilon$ . However, NOMP has been able to perfectly recover all signals with a window size of  $w = 6$ .

Another advantage to NOMP is its fast runtime. Its runtime is comparable to that of OMP because NOMP only alters the way OMP selects the support set. SSCoSaMP is much slower due to its need to run another traditional CS algorithm in its identify and prune steps. More importantly, NOMP succeeds on a much wider range of signal structures. Thus, if the signal structure is unknown or the class of signals of interest includes many types, NOMP offers many practical advantages.



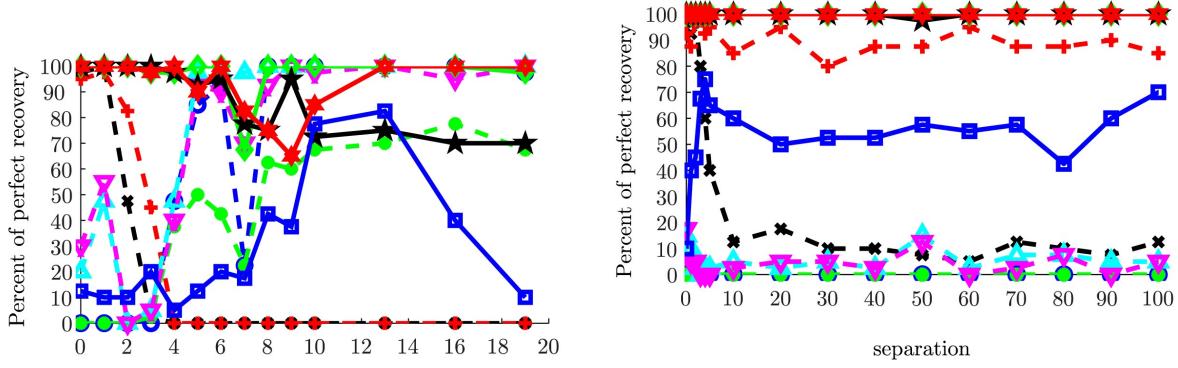


Figure 6: Left: Percent of perfect recovery of a well-separated signal as a function of the separation between nonzero coefficients. Legend given in Figure 5. We measure the percent of perfect recovery as the separation between each nonzero coefficient increases. Right: Percent of perfect recovery of a signal with two clusters as a function of the distance between clusters. Legend given in Figure 5. We measure the percent of perfect recovery as the separation between the two clusters increases.

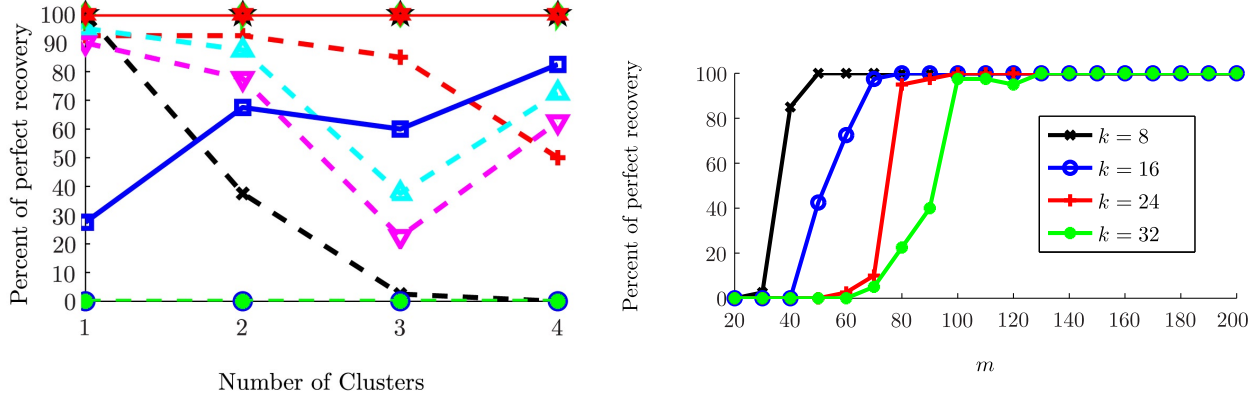


Figure 7: Left: Percent recovery of a signal with one, two, three and four clusters. Legend given in Figure 6. Right: Percent recovery of NOMP on clustered signals with varying levels of sparsity.

## 4.2 USSCoSaMP

The results of Figure 1 suggest that one class of algorithms work well for the clustered signal model and a disjoint set of algorithms work well for the well-separated model. Indeed, when OMP and  $\ell_1$ -minimization are used to form these projections, SS-CoSaMP is able to accurately recover the signal when the nonzero entries in the coefficient vector are well-separated, but unable to recover the signal when the nonzeros are clustered together. Conversely, when CoSaMP is used to form these projections, SS-CoSaMP exhibits near-perfect signal recovery when the coefficient vector's nonzeros are clustered, and is unable to recover the signal when the nonzeros are well-separated.

A natural question is whether one can leverage the benefits of each algorithm type simultaneously in order to obtain successful reconstruction of both signal types, as well as those in between. For computational efficiency, we choose to focus on the greedy algorithms. After observing the signal recovery success of OMP and CoSaMP in the two “extreme” cases of coefficient vector structure, we seek to combine their performance to create a method that can succeed at both extremes and everywhere in between.

A natural way to combine the recovery performance of SS-CoSaMP (OMP) and SS-CoSaMP (CoSaMP) results in the USSCoSaMP algorithm (acrostic: Unionized SS-CoSaMP) which is detailed in Algorithm 3.

USSCoSaMP's crucial improvement over SS-CoSaMP appears in the prune step. SS-CoSaMP makes use of a simple CS algorithm, classically either OMP or CoSaMP, to determine the best  $k$ -sparse approximation of  $\alpha$ , de-



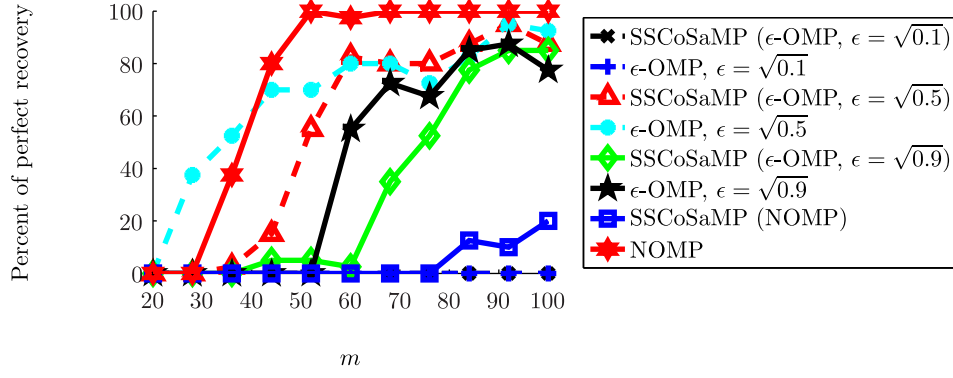


Figure 8: Percent of perfect recovery of NOMP and  $\epsilon$ -OMP for various choices of  $\epsilon$  on hybrid signals.

---

**Algorithm 3** Unionized Signal-Space CoSaMP (USSCoSaMP)

---

**Input:**  $A, D, y, k$ , stopping criterion  
**Initialize:**  $r = y, x_0 = 0, \ell = 0, \Gamma_{old} = \emptyset$   
**while** not converged **do**  
    **Proxy:**  $\tilde{v} = A^* r$   
    **Identify:**  $\Omega = \mathcal{S}_D(\tilde{v}, 2k)$   
    **Merge:**  $T = \Omega \cup \Gamma_{old}$   
    **Update:**  $\tilde{w} = \arg\min_z \|y - Az\|_2 \quad \text{s.t.} \quad z \in \mathcal{R}(D_T)$   
    **Prune:**  $\Gamma_{omp} = \mathcal{S}_D(\tilde{w}, k)$   
               $\Gamma_{cosamp} = \mathcal{S}_D(\tilde{w}, k)$   
    **Union:**  $\Gamma = \Gamma_{omp} \cup \Gamma_{cosamp}$   
    **Update:**  $\tilde{x} = \arg\min_z \|y - Az\|_2 \quad \text{s.t.} \quad z \in \mathcal{R}(D_\Gamma)$   
               $x_{\ell+1} = \mathcal{P}_T \tilde{x}$   
               $r = y - Ax_{\ell+1}$   
               $\ell = \ell + 1$   
               $\Gamma_{old} = \Gamma$   
**end while**  
**Output:**  $\hat{x} = x_\ell$

---

noted  $\hat{\alpha}$ , and returns both  $\hat{\alpha}$  and  $\text{supp}(\hat{\alpha})$ . By contrast, USSCoSaMP uses both OMP and CoSaMP to compute the best  $k$ -sparse approximation of  $\alpha$ , and only returns the supports of each approximation, denoted in Algorithm 3 as  $\Gamma_{omp}$  and  $\Gamma_{cosamp}$ . After taking the union of these supports,  $\Gamma$ , USSCoSaMP performs the same final step as OMP and CoSaMP by solving a least squares problem restricted to the “best” columns. Because  $k \leq |\Gamma| \leq 2k$ , the  $\hat{\alpha}$  returned by USSCoSaMP is between  $k$ - and  $2k$ -sparse.

Like SSSCoSaMP, there are different variants of USSCoSaMP. The key change between these variants lies in the identification step. We focus on two variants that perform the best over the broadest class of sparse coefficient vectors. The first alternates between using OMP and CoSaMP to compute  $\Omega$ . In this case,  $|\Omega| = 2k$  always. The second has OMP and CoSaMP compute  $\Omega_{omp}$  and  $\Omega_{cosamp}$  respectively, then sets  $\Omega = \Omega_{omp} \cup \Omega_{cosamp}$ . In this case,  $2k \leq |\Omega| \leq 4k$ . In the following figures, these variants are denoted USSCoSaMP (alt) and USSCoSaMP (union) respectively.

Figure 9 shows the recovery performance of USSCoSaMP variants benchmarked against the traditional SSSCoSaMP variants for nonzero entries in the coefficient vector that are clustered, well-separated, and hybrid, respectively. For the first two experiments,  $k = 8, n = 256$ , and  $d = 1024$ . The number of measurements  $m$  was incremented from 20 to 100 by 5, and 100 trials were conducted for each value of  $m$  (500 trials for the hybrid signal case).

Figure 9 (top) shows both USSCoSaMP variants performing as well as the excellent SSSCoSaMP (CoSaMP)

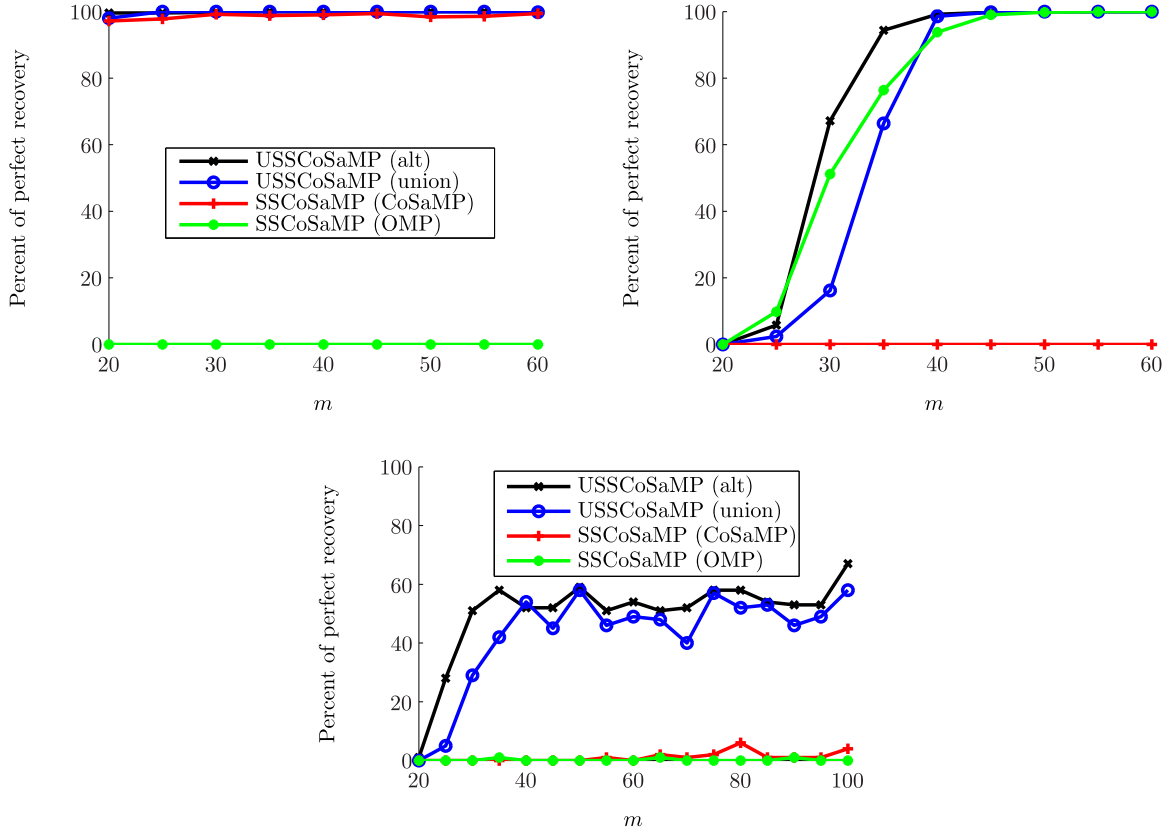


Figure 9: USSCoSaMP and SSCoSaMP variants recovery performance on sparse clustered signals (left), sparse well-separated signals (right), and hybrid sparse signals (bottom). USSCoSaMP (alt) alternates between using OMP and CoSaMP to compute  $\Omega$  whereas USSCoSaMP (union) utilizes both simultaneously and computes  $\Omega$  as the union of the two output support sets.

in the clustered case, boasting near perfect recovery even for low values of  $m$ . Figure 9 (middle) shows both USSCoSaMP variants performing as well as SSCoSaMP (OMP) for  $m \geq 40$  in the well-separated case. In fact, the alternating USSCoSaMP variant even outperforms SSCoSaMP (OMP) in the  $m \geq 30$  range, requiring only  $m = 40$  for 100% recovery versus  $m = 50$  for SSCoSaMP (OMP). This latter observation is surprising, since SSCoSaMP (OMP) already performs well on this signal model whereas SSCoSaMP (CoSaMP) performs quite poorly. Figure 9 (bottom) shows USSCoSaMP sporting a modest recovery performance in the hybrid case, a marked improvement over both SSCoSaMP variants, although still not reaching 100% recovery. We do note that in the trials where the signal was not accurately recovered, the residual also never reached the lower threshold, so at least one can detect when such a failure has occurred. This is the only generalized method to our knowledge that performs well on both clustered and separated models simultaneously. Together, the figures show USSCoSaMP performing better than the sum of its parts.

## 5 Conclusion

In our work, we conduct a rigorous empirical investigation into the recovery performance of  $\ell_1$ , OMP, CoSaMP and SSCoSaMP on different types of sparse signals. We summarize the results of our experiments in Table 1. The values in the table represent the percent of perfect recovery at  $m = 100$  measurements. The red values are between 0-39%, the black represent 40-79%, and the blue a range of 80-100%. We see that the best performers over the broadest class of signals are USSCoSaMP and NOMP.

The new algorithms that we developed, NOMP and USSCoSaMP, improve upon existing algorithms for recovering signals in a DFT dictionary. Both bridge the gap between clustered and well-separated signal types and are

Method	Clustered	Spread	Hybrid	Two Cluster	Four Cluster	Alternating	Pair Spread
SSCoSaMP (CoSaMP)	100	0	0	20	0	100	0
SSCoSaMP ( $\ell_1$ )	20	100	30	10	60	25	35
SSCoSaMP (OMP)	0	100	0	0	0	0	5
CoSaMP	100	0	10	100	60	100	0
OMP	0	60	0	0	0	0	10
$\ell_1$	20	100	20	10	50	20	25
USSCoSaMP	100	100	65	80	30	100	10
NOMP	100	100	100	100	100	100	100

Table 1: SSSCoSaMP variants and new algorithms’ performance on various types of sparse coefficient vectors. All of which are sparse with respect to a  $4\times$  overcomplete DFT dictionary. A minimum of 40 trials were performed on each test.

able to give high recovery on other types of signals as well. Our experimental results provide more insight on the subject and can serve as a guide for practitioners using these methods. NOMP works well for basically all signal types and has a fast runtime, but is specific to certain dictionary structures. USSCoSaMP also provides accurate recovery when the signal is well-separated or clustered and often on hybrid signals. We thus provide a catalog of existing and new algorithms along with their performance on different signal structures. It is important future work to continue the development and analysis of efficient methods that provide accurate recovery of signals in the framework of arbitrary dictionaries.

## References

- [1] T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. *Appl. Comput. Harmon. A.*, 27(3):265–274, 2009.
- [2] E. J. Candès, Y. C. Eldar, D. Needell, and P. Randall. Compressed sensing with coherent and redundant dictionaries. *Appl. Comput. Harmon. A.*, 31(1):59–73, 2010.
- [3] E. J. Candès and C. Fernandez-Granda. Towards a mathematical theory of super-resolution. *Commun. Pur. Appl. Math.*, 67(6):906–956, 2014.
- [4] E. J. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Commun. Pur. Appl. Math.*, 59(8):1207–1223, 2006.
- [5] E. J. Candès, M. Rudelson, R. Vershynin, and T. Tao. Error correction via linear programming. *46th Annual Symp. Found. Computer Science*, pages 668–681, 2005.
- [6] E. J. Candès and T. Tao. Decoding by linear programming. *IEEE T. Inform. Theory*, 51:4203–4215, 2005.
- [7] M. Davenport, D. Needell, and M. B. Wakin. Signal space CoSaMP for sparse recovery with redundant dictionaries. *IEEE T. Inform. Theory*, 59(10):6820, 2012.
- [8] M. A. Davenport and M. B. Wakin. Compressive sensing of analog signals using discrete prolate spheroidal sequences. *Appl. Comput. Harmon. A.*, 2011. To appear.
- [9] R. Giryes and M. Elad. OMP performance with highly coherent dictionaries. *Proc. Int. Conf. on Sampling Theory and Appl.*, 2013.
- [10] R. Giryes and D. Needell. Greedy signal space methods for incoherence and beyond, 2013. Submitted.
- [11] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE T. Signal Process.*, 41(12):3397–3415, 1993.

- [12] D. Needell and J. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. A.*, 26(3):301–321, 2009.
- [13] D. Needell and R. Vershynin. Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *IEEE J. Sel. Top. Signa.*, 4:310–316, 2007.
- [14] H. Rauhut, J. Romberg, and J. A. Tropp. Restricted isometries for partial random circulant matrices. *Appl. Comput. Harmon. A.*, 32(2):242–254, 2012.
- [15] M. Rudelson and R. Vershynin. On sparse reconstruction from fourier and gaussian measurements. *Comm. Pure Appl. Math.*, 61:1025–1045, 2008.
- [16] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via Orthogonal Matching Pursuit. *IEEE T. Inform. Theory*, 53(12):4655–4666, 2007.
- [17] T. Zhang. Sparse recovery with orthogonal matching pursuit under RIP. *IEEE T. Inform. Theory*, 57(9):6215–6221, 2011.